IN THE UNITED STATES DISTRICT COURT
FOR THE WESTERN DISTRICT OF TEXAS
MIDLAND-ODESSA DIVISION

MALIKIE INNOVATIONS LTD., KEY PATENT INNOVATIONS LTD. Plaintiffs, v. MARA HOLDINGS, INC. (F/K/A MARATHON DIGITAL HOLDINGS, INC Defendant.	CASE NO. 7:25-cv-00222-DC-DTG JURY TRIAL DEMANDED

MARA'S OPENING CLAIM CONSTRUCTION BRIEF

TABLE OF CONTENTS

TABLE OF AUTHORITIES

Page(s)

CASES

Defendant MARA Holdings, Inc. ("MARA") respectfully submits its Opening Claim Construction Brief regarding disputed terms in U.S. Patent Nos. 8,532,286 ("the '286 Patent"), 7,372,960 ("the '960 Patent"), 8,666,062 ("the '062 Patent"), 8,788,827 ("the '827 Patent"), 10,284,370 ("the '370 Patent"), and 7,372,961 ("the '961 Patent").

I. TECHNICAL BACKGROUND

A. Modular Arithmetic and Reduction

Modular arithmetic is a type of arithmetic performed on integers where the numbers "wrap around" a certain number called the **modulus**, which is often denoted as *n*. One analogy is how hours work on a standard clock having 12 hours. If it is 9:00 now and 5 hours pass, the hour hand points to 2:00 (not 14:00). In ordinary arithmetic, 5 + 9 = 14; but in the clock example, 5 + 9 = 2 because the numbers "wrap around" 12. There, the modulus is 12, and the operation of adding 5 to 9 is written as: 5 + 9 mod 12 = 2. Ex. A, ¶24.

1. Reduction

"Modular reduction" or "reduction," commonly represented by the "mod" operation, is used to reduce the value of a number larger than a given modulus to a value below that modulus. Ex. A, ¶25. Given a modulus *n*, the integers [0, 1, 2 … *n* – *1*] are the canonical residues of *n*. *Id.* An integer *a* that results from an arithmetic operation and that is not within the set of [0, 1, 2 … *n* – *1*] may be "reduced" back to a canonical residue by dividing *a* by the modulus *n* and outputting the remainder. *Id*. This operation is commonly denoted as *a* mod *n*. *Id.* For example, 7 mod 2 = 1 and 10 mod 4 = 2.

The "≡" sign denotes "congruent to" or "modularly equivalent to," which means that the two numbers are equivalent given a certain modulus. *Id.*, ¶¶26-27. More formally, two numbers are congruent mod *n*, if they have the same remainder when divided by *n*. For example, 38, 26, 14, and 2 are congruent to each other mod 12 because each of them, when divided by 12, has a

remainder of 2. *Id.* Notably, two numbers congruent to each other mod *n* are interchangeable in the mod *n* system. *Id.*

2. Arithmetic

In general, the goal modular of arithmetic is to compute *a* op *b* mod *n*, where "op" denotes an arithmetic operation. Ex. A, ¶¶28-29. For example, modular addition calculates $a + b \bmod n$ and modular multiplication calculates $a \times b \bmod n$. *Id.* These operations can be performed simply by first performing ordinary addition, subtraction, or multiplication, followed by modular reduction.[1] *Id.* For example:

$$4 + 5 \bmod 7 = 9 \bmod 7 = 2$$

$$4 - 5 \bmod 7 = -1 \bmod 7 = 6$$

$$4 \times 5 \bmod 7 = 20 \bmod 7 = 6$$

Id., ¶29.

One fundamental theorem of modular arithmetic is that, for addition, subtraction, and multiplication, "reducing each intermediate result" with the modulus *n* "gives the same answer as computing in ordinary integer arithmetic and reducing the result mod *n*." Ex. U at 72-73; Ex. A, ¶¶31-33. For example, one may calculate 25 multiplied by 15 with a modulus of 97 in the following way: $25 \times 15 \; mod \; 97 = 375 \; mod \; 97 = 84$. Ex. A, ¶32. Alternatively, using the theorem, the calculation can be performed in this way yielding the same result:

$$(25 \times 15) \; mod \; 97$$

$$= [(20 + 5) \times (10 + 5)] \; mod \; 97$$

$$= [200 + 100 + 50 + 25] \; mod \; 97$$

$$= [200 \; mod \; 97 + 100 \; mod \; 97 + 50 \; mod \; 97 + 25 \; mod \; 97] \; mod \; 97$$

[1] Modular inversion and division are less similar to ordinary inversion and division because there are no fractions in modular arithmetic. Ex. A, ¶30.

$$= [6 + 3 + 50 + 25] \; mod \; 97$$

$$= 84$$

Id. The latter method, although appearing to involve more steps on paper, is advantageous in computer-implemented methods, which perform calculations in binary numbers. *Id*, ¶33.

B. Finite Fields and Computer Implementation

Modular arithmetic is particularly useful in cryptography because many cryptographic systems rely on finite fields. Ex. A, ¶¶34-35. A finite field is a field having a finite number of elements. One type of finite field widely used in cryptography is a called a "prime field" (commonly denoted "F_p"), which is a field having a prime number of field elements that are consecutive integers starting from 0 (i.e., a prime field having 97 elements consists of integers 0 to 96).[2] *Id.* For such a finite field, the number of elements is defined as the modulus of the field (e.g., n = 97). *Id.* Performing operations on the elements of the finite field should yield an output that is also within the field. *Id*. Accordingly, modular arithmetic is used to perform arithmetic in finite fields. *Id*.

In computers, finite field elements and finite field moduli are represented in binary. *Id.*, ¶¶36-40. Each computer processor has its native "word" size, which is a number of binary bits the processor can handle as a unit. *Id.* The word-size typically corresponds to the processor's register width and is commonly indicated in the "n-bit" designation of the processor. For example, a "32-bit" processor would have a 32-bit word size. *Id.* A processor can perform word-sized operations most efficiently, such as adding two 32-bit integers on a 32-bit processor. *Id*. Performing the same type of operation on data larger than the native word size typically takes much longer because it requires multiple instructions. *Id*.

[2] Another type of finite field is a binary extension field (denoted by F_{2^n}). Ex. A, ¶35.

Computer implementation of finite-field-based cryptography typically requires a large finite field, and therefore, the elements of the finite field are much longer than the processor's word size (typically 64-bit or less). *Id*. To facilitate easier and faster computation, finite field elements are typically broken into and stored in multiple machine words. *Id*. A common way to determine the number of words required to store a finite field element is to divide the bit-length of the finite field modulus with the word size of the processor, then round up an integer. *Id*. For example, assume use of a 32-bit processor with a prime field F_{521} whose modulus is $2^{521} - 1$ (this is a prime number). *Id*. The length of the modulus is 521 bits and the word size of 32 bits. *Id*. Therefore, 17 words (521 ÷ 32, rounded up) are required to store a field element a. *Id.* A common way to denote the word-sized representation of a is $a = (a_{16}, \ldots a_1, a_0)$. In this example, the right-most word (a_0) is the least significant word ("LSW"), while the left-most word (a_{16}) is the most significant word. *Id*. As an example in decimal, assume a word size of 3, then the number 123,456,789 broken into words is $a_2 = 123$ (the MSW), $a_1 = 456$, and $a_0 = 789$ (the LSW). *Id.*

C. Methods for Performing Modular Reduction in Cryptography

As explained above, performing finite field arithmetic involves modular reduction, which involves dividing an input by the modulus and calculating the remainder. Ex. A, ¶41. However, performing division on a computer can be computationally difficult and expensive. *Id*. Therefore, many methods for optimizing the calculation of modular reduction (i.e., calculating a mod n) have been developed. *Id.* Two methods relevant to the claim construction disputes in this case are explained below. *Id.*

1. Montgomery Reduction

Montgomery reduction was originally proposed by Peter Montgomery in 1985. Ex. A, ¶42. A defining characteristic of Montgomery reduction is that it reduces "from below," meaning it proceeds by clearing the least significant portions of the unreduced quantity, leaving the

remainder in the upper portion. *Id.*, ¶45. Montgomery reduction uses a special value R, which is typically a large power of 2. *Id.*, ¶¶43-44. To calculate *a* mod *n* using Montgomery reduction, the operand *a* is converted to the Montgomery form (T = *aR* mod *n*). *Id.* Next, an integer multiple of the modulus *n* is added (i.e., $T' = T + m \times n$) such that the least significant *k* bits of the sum will be zero. *Id.* The sum *T'* can then be shifted to the right to drop the zeroed least significant *k* bits. *Id*. The process is repeated until the desired reduction is achieved. *Id.*

An example in decimal is provided below to illustrate how Montgomery reduction is used to calculate **4355 mod 97 = 87**. *Id.*, ¶¶46-47. In this example, the radix is a power of 10, because the example is in decimal (not binary). *Id.*

Montgomery Reduction
Assumptions: modulus n = 97; word-size = 2; a = 4355; Montgomery radix *R* = 10000 **Goal:** calculate 4355 mod 97 (i.e., a mod n) **Answer:** 4355 mod 97 = 87
1. Convert a to Montgomery form (T): $T = a \times R \bmod n = 4355 \times 10000 \bmod 97 \equiv 39195$ **2. Calculate μ and m:** $\mu = (-n)^{-1} \bmod 10^w = (-97)^{-1} \bmod 100 = 67$ $m = \mu \times T_0 \bmod 10^w = 67 \times 95 \bmod 100 = 65$ **3. Add m × n to T such that the LSW (the last two digits) is zero:** $39195 + 65 \times 97 = 45500$ **4. Shift T to the right by 2 digits:** $45500 \rightarrow 455$ **5. Calculate m** $m = \mu \times T_0 \bmod 10^w = 67 \times 55 \bmod 100 = 85$ **6. Add m × n to T such that the LSW (the last two digits) is zero:** $455 + 85 \times 97 = 8700$ **7. Shift T to the right by 2 digits:** $8700 \rightarrow 87$

2. Pseudo-Mersenne Reduction

Another modular reduction method commonly used in cryptography is pseudo-Mersenne reduction, disclosed by Richard Crandall in U.S. Patent No. 5,159,632 issued in 1992 and published in an article from 2006, *Efficient Software-Implementation of Finite Field with*

Applications to Cryptography, Springer Science, Guajardo, Jorge et al. Ex. A, ¶48. Unlike Montgomery reduction, pseudo-Mersenne reduction reduces from above—i.e., targets the most significant word(s) rather than the least significant word. *Id.*, ¶50. A detailed description of pseudo-Mersenne reduction is included in the declaration of Dr. Koc. *Id.*, ¶¶48-50. An example in decimal is provided below to illustrate how pseudo-Mersenne reduction is used to calculate **4355 mod 97 = 87**. *Id.*, ¶¶49-50.

Pseudo-Mersenne Reduction
Assumptions: modulus n = 97; word-size = 2; a = 4355 **Goal:** calculate 4355 mod 97 (i.e., a mod n) **Answer:** 4355 mod 97 = 87
1. Obtain c, where n = 10^k – c $97 = 10^2 - 3$ c = 3 **2. Split a = 4355 into a high word (h) and a low word (l):** h = 43; l = 55 **3. Calculate a mod n = h × c + l:** 43 × 3 + 55 = 184 **4. Split t = 184 into a high word (h) and a low word (l):** h = 1; l = 84 **5. Calculate t mod n = h × c + l:** 1 × 3 + 84 = 87

II. DISPUTED CLAIM TERMS

A. The ’286 Patent

The ’286 Patent is directed to a modified version of the prior art Montgomery reduction. Ex. B at 1:13-16 (“The following…provides a system and method for reducing the computation and storage requirements for a Montgomery-style reduction.”). The purported invention is to “produce a Montgomery reduction…by storing a new precomputed value used to substantially replace the μ and n values used in Montgomery reduction with a single value.” *Id*. at 3:27-31. Specifically, a “new value” **$n' = 2^{-w} \bmod n$** is “used to zero the least significant non-zero word of a at each iteration, without the need to first multiply by μ and determine m.” *Id.* at 5: 45-49. The

allegedly inventive method is described as follows:

> To be explicit $a \equiv [. . . , a_4, a_3, a_2, a_1, a_0]$ is replaced with $a \equiv [. . . , a_4, a_3, a_2, a_1, 0] + a_0 \times n' \times 2^w$. Since $a_0 \times n' \times 2^w$, taken without reduction, is zero in its least significant digit (by the shift 2^w), the resulting value is 0 in its least significant digit, which is the desired low-order reduction. Typically this zero digit will be treated by shifting (either logically or physically) the value down by a digit. The modified reduction value $n' = 2^{-w} \bmod n$, is a convenient replacement for the values μ and n used in the Montgomery reduction method.

Id. at 5:60-6:2.

The '286 Patent illustrates an example of this process in Figure 7. *Id*. at 6:32-34 ("As shown in FIG. 7, by obtaining and using the modified reduction value n′ and applying the relationship $a \equiv [. . . , a_4, a_3, a_2, a_1, 0] + a_0 \times n' \times 2^w$ at each iteration, the least significant word of a is zeroed and the remaining words modified."). As shown in Figure 7, the input **a** consists of 10 words **(a_9, … a_0)**. The allegedly inventive method teaches setting **a_0** (the LSW of **a**) to zero and adding **$a_0 \times n' \times 2^w$** to **a**, where **$n' = 2^{-w} \bmod n$**.[3] Ex. A, ¶55. Because the LSW of the addend **$a_0 \times n' \times 2^w$** is also zero, the LSW of post-addition **a** is now zero. *Id.*; Ex. B at 5:60-6:2 ("$a_0 \times n' \times 2^w$, taken without reduction, is zero in its least significant digit."). One may then shift **a** to the right by one word to take the empty space of the LSW, thus reducing the length of **a**. Ex. A, ¶55. The process is repeated a number of times until the desired amount of reduction is achieved to produce an output that is **$aR^{-1} \bmod n$**. *Id*. The '286 Patent discloses that its method cannot be used for the last iteration, and the standard prior art Montgomery method must be used. Ex. B at 6:44-50, 9:1-2 ("Now using standard Montgomery reduction for the last digit.").

An example in decimal is provided below to illustrate the '286 Patent's modified Montgomery-style method shown in Figure 7. Ex. A, ¶56. Note that the '286 Patent method (Fig.

[3] Note that these two steps can be performed in either order without affecting the result. Ex. A, ¶55 n.2.

7) adds $a_0 \times n' \times \mathbf{2^w}$ but the example below using the '286 Patent method adds $a_0 \times n' \times \mathbf{10^w}$. This is because the example below is performed in decimal instead of binary, therefore the base is 10 instead of 2. *Id*.

'286 Patent Method (Fig. 7)
Assumptions: modulus n = 97; word-size = 2; a = 4355 **Goal:** calculate 4355 mod 97 (i.e., a mod n) **Answer:** 4355 mod 97 = 87
1. Convert t to Montgomery form (T): $T = a \times R \bmod n = 4355 \times 10000 \bmod 97 \equiv 39195$ **2. Calculate n' = 10^{-w} mod n:** $n' = 10^{-2} \bmod 97$ $n' = 65$ **3. Zero the LSW of T** $39195 \rightarrow 39100$ **4. Add $T_0 \times n' \times 10^w$ to T:** $39100 + 95 \times 65 \times 10^2 = 656600$ **5. Shift T to the right by two digits:** $656600 \rightarrow 6566$ **Perform standard Montgomery reduction on 6566:** **6. Calculate μ and m:** $\mu = (-n)^{-1} \bmod 10^w = (-97)^{-1} \bmod 100 = 67$ $m = \mu \times T_0 \bmod 10^w = 67 \times 66 \bmod 100 = 22$ **7. Add m × n to T such that the last two digits are zero:** $6566 + 22 \times 97 = 8700$ **8. Shift T to the right by 2 digits:** $8700 \rightarrow 87$

1. "Montgomery-style reduction"

MARA's Construction	Malikie's Construction
"reduction that proceeds by clearing the least significant portions of an unreduced operand and leaving the remainder in the more significant portions"	This term appears only in the preamble and is not limiting. Should the Court find this term is limiting, it should be construed to mean: "reduction that proceeds by clearing the least significant portions of an unreduced quantity, leaving the remainder in the more significant portions."

A preamble may be limiting if it, *inter alia*, "is essential to understand limitations or terms in the claim body" or "recites additional structure or steps underscored as important by the specification." *Georgetown Rail Equipment Company* v. *Holland L.P.*, 867 F.3d 1229, 1236 (Fed. Cir. 2017). Specifically, the Federal Circuit has found that "the preamble is limiting when [it] tethers the claim to the focus of the described invention," which "is not understood solely from body of the claim." *UNILOC 2017 LLC* v. *Verizon Comm'n, Inc.*, 2020 WL 805271 (E.D. Tex. Feb. 18, 2020) (citing cases).

For example, in *Corning Glass Works* v. *Sumitomo Elec. U.S.A., Inc.*, the preamble recited "an optical waveguide," while the claim body was not limited to optical fibers having waveguide properties. 868 F.2d 1251, 1256-57 (Fed Cir. 1989). The Federal Circuit found the preamble limiting because the "specification ma[de] clear that the inventors were working on the particular problem of an effective optical communication system not on general improvements in conventional optical fibers." *Id.* The Federal Circuit reasoned that "read[ing] the claim in light of the specification indiscriminately to cover all types of optical fibers would be divorced from reality" when "[t]he invention is restricted to those fibers that work as waveguides as defined in the specification." *Id.* The preamble therefore gave "life and meaning and provide[d] further positive limitations to the invention claimed." *Id.*

Similarly, in *GE Co.* v. *Nintendo Co.*, the preamble recited a "system for displaying a pattern on a ***raster scanned*** display device" but the claim body was directed to all types of display systems. 179 F.3d 1350, 1361–62 (Fed. Cir. 1999). The Federal Circuit found the preamble limiting because the "specification ma[de] clear that the inventors were working on the particular problem of displaying binary data on a raster scan display device and not general improvements to all display systems." *Id.* As in *Corning Glass*, it found that "read[ing] the claim indiscriminately to cover all types of display systems would be divorced from reality" when the "invention so described is restricted to those display devices that work by displaying bits[.]" *Id.*

The term "Montgomery-style reduction," which appears only in the preamble of claim 1, is limiting under the above cases. The specification and prosecution history are clear that the invention is directed to a particular improvement in Montgomery reduction, not a general improvement to any type of reduction. Ex. B at 1:13-16 ("The following … provides a system and method for reducing the computation and storage requirements for a ***Montgomery-style reduction***."), 3:20-26 ("To ***improve the reduction efficiency of a Montgomery machine***, the objective should be to reduce the number of operations…"), 3:27-29 ("In the following embodiments, a system and method are utilized that ***provide an alternative way in which to produce a Montgomery reduction from below***[.]"), 3:40-42 ("By ***modifying the Montgomery reduction mechanism*** in this way, the number of multiplications and registers required to effect the Montgomery reduction can be reduced."), Abstract ("A system and method are described that ***provide an alternative way in which to produce a Montgomery reduction*** from below"); Title ("System and method for reducing the computation and storage requirements for a ***Montgomery-style reduction***"); *see also* Ex. H at 1402-3 (describing "present application" as "provid[ing] ***an alternative way*** in which to produce a ***Montgomery reduction from below***); *id.* ("By ***modifying***

the Montgomery reduction mechanism in this way, the number of multiplications and registers required to effect the Montgomery reduction can be reduced."), *id.* (describing claimed method as "***this modified Montgomery reduction***").

Malikie's position, however, attempts to read the claim body in isolation from the specification, and thereby covers reduction operations that are not Montgomery-style and do not reduce from below, such as the prior art pseudo-Mersenne reduction (*supra*, I.C.2), which reduces from above. Ex. A, ¶58-64. Reading the claim as Malikie proposes to indiscriminately cover these other types of reduction methods would be divorced from reality. Accordingly, the term "Montgomery-style reduction" gives "life and meaning and provides further positive limitations to the invention claimed." *Corning Glass*, 868 F.2d 1257.

If the Court finds the preamble limiting, only a small dispute remains as to the meaning of "Montgomery-style reduction." Both constructions are pulled from the specification at column 11, lines 44-46, and are nearly identical except that MARA's construction recites "operand" while Malikie's construction recites "quantity." MARA's construction should be adopted because it is consistent with the claim language, which requires that the "operand" is the unreduced quantity on which the reduction proceeds. Ex. B at 9:26-27 (claim language requiring "obtaining an operand" and "computing a modified operand."). The claims do not recite "quantity" as Malikie proposes.

2. "perform a replacement of a least significant word of the operand"

MARA's Construction	Malikie's Construction
"add a modular equivalent of the operand's least significant word to the more significant words of the operand such that the result can be shifted down to drop the least significant word"	"replace a word that makes the smallest contribution to the value of the operand"

The parties agree that "least significant word of the operand" from the claim term is a word that makes the smallest contribution to the value of the operand but disagree as to what it means

to "perform a replacement." Malikie's construction provides no definition for the term, and merely states that "perform a replacement" means "replace". Malikie's Response to MARA's Renewed Motion to Dismiss indicates that Malikie understands "perform a replacement" to cover any instance involving setting the value of the LSW of the operand. *See* Dkt. 41 at 5 (alleging that accused instrumentalities infringe because "each of the words of 'r' … are set one at a time, starting with the [LSW] 'd[0],'" and "setting 'd[0]' replaces the [LSW] of the operand."). While "cancellation" is a term known in the art with a particular meaning in the context of Montgomery reduction, the term "replacement" was not previously used in the art and instead was a term the inventors used to characterize their new method of Montgomery-style reduction, which does not use cancellation. Ex. A, ¶¶51-54. The Court should adopt MARA's construction, which is consistent with the surrounding claim language and the specification and reject Malikie's which is divorced from both.

First, the claim language requires performing a "Montgomery-style reduction by "computing a modified operand … to perform a replacement of a least significant word of the operand, ***rather than perform a cancellation thereof***." That is, the surrounding claim language requires performing Montgomery-style reduction by replacement rather than cancellation and thereby requires that the two are distinct techniques for achieving the same end. The examples provided in Sections I.C.1 and II.A illustrate this succinctly by showing how both standard Montgomery method (cancellation) and the '286 Patent method (replacement) are used to compute 4355 mod 97. While both methods involve clearing the least significant portions of an unreduced operand and leaving the remainder in the more significant portions, each does so in mathematically distinct ways as reflected in MARA's constructions of both terms. Specifically, cancellation proceeds by adding a multiple of the modulus **n** (e.g., **m** × **n**) to the operand **T**, and the multiple

of the modulus is calculated in such that the LSW of the resulting sum is always zero, allowing one to shift the result down to drop the LSW. *Infra*, II.A.3. In contrast, replacement proceeds by adding a modular equivalent of the LSW (e.g., **LSW×n′×10^w**) to the operand **T**, which also allows one to shift the result down to drop the LSW. The pre-shift and shifted results of each are modularly equivalent to one another and therefore interchangeable, ultimately resulting in the correct calculation of 4355 mod 95. Ex. A, ¶57. Accordingly, MARA's construction is consistent with the claim language because it distinguishes replacement from cancellation.

Second, MARA's construction is consistent with, but not limited to, the only embodiment in the specification, which explicitly teaches a POSA what a replacement means (Ex. A, ¶54):

> To see the usefulness of this new value, it is noted that if the value n′ is then shifted up by one digit, which is equivalent to multiplying by 2^w, a value is obtained that is equivalent to 1 mod n. Consequently, the value a_0 can be replaced with $a_0 \times n' \times 2^w$, that is, a_0 multiplied by n′ shifted up one digit. **To be explicit $a \equiv [\ldots, a_4, a_3, a_2, a_1, a_0]$ is replaced with $a \equiv [\ldots, a_4, a_3, a_2, a_1, 0] + a_0 \times n' \times 2^w$.** Since $a_0 \times n' \times 2^w$, taken without reduction, is zero in its least significant digit (by the shift 2^w), the resulting value is 0 in its least significant digit, which is the desired low-order reduction. Typically this zero digit will be treated by shifting (either logically or physically) the value down by a digit.

Ex. B at 5:59-67. MARA's construction tracks this disclosure. Specifically, the specification provides an operand **a** with LSW **a_0**, and discloses performing replacement by zeroing **a_0** and adding a modular equivalent of **a_0** (i.e., **$a_0 \times n' \times 2^w$**) to the remaining words. The specification further describes that, following those steps, the result can be shifted down by one word. *Id.*; *see also* Fig. 7; *id.* at 3:32-38 ("As show in Fig. 7, by … applying the relationship $a \equiv [\ldots, a^4, a^3, a^2, a^1, 0] + a^0 x n' x 2^w]$ at each iteration, the [LSW] of a is zeroed and the remaining words modified. As before, either during or after this operation, the modified words are shifted down such that the next [MSW] becomes the next [LSW] for the next iteration."). MARA's construction encapsulates each of these steps without improperly narrowing the term to this exact embodiment.

In contrast, Malikie's overbroad construction does not track the specification whatsoever. *See Phonometrics* v. *N. Telecom*, 133 F.3d 1459, 1466 (Fed. Cir. 1998) ("Although claims are not necessarily restricted in scope to what is shown in a preferred embodiment, neither are the specifics of the preferred embodiment irrelevant to the correct meaning of claim limitations."). Indeed, Malikie's interpretation of the claim is so overbroad and divorced from the specification, that under its constructions, the claim reads directly on the prior art pseudo-Mersenne reduction. Ex. A, ¶¶58-64.

3. "perform a cancellation thereof"

MARA's Construction	Malikie's Construction
"add a multiple of the modulus to the operand such that the least significant word of the result is zero and the result can be shifted down to drop the least significant word"	"add a multiple of the modulus to the operand to eliminate the least significant word of the operand"

The parties agree that this term involves adding a multiple of the modulus, but disagree as to whether the addition is done "such that the [LSW] of the result is zero and the result can be shifting down to drop the [LSW]" or whether the addition is done "to eliminate the [LSW] of the operand." The Court should adopt MARA's construction because it is how the term cancellation is described in the '286 Patent and how a POSA would understand the term in the context of Montgomery reduction.

First, as explained *supra* II.A.2, MARA's construction is consistent with the claim language because it distinguishes cancellation from replacement as a mathematically distinct technique for performing Montgomery-style reduction.

Second, the specification is clear that "cancellation" is the way in which standard Montgomery reduction is performed (while "replacement" is the modified way proposed by the '286 Patent). Ex. B at 3:35-38 ("rather than cancellation thereof, as performed in standard

Montgomery reduction."). A POSA would understand that standard Montgomery reduction involves adding a multiple of the modulus to the operand such that the LSW of the result becomes zero and the LSW can be dropped by shifting. Ex. A, ¶51. This is the meaning of "cancellation," and is precisely how the '286 Patent describes this step in standard Montgomery reduction:

> ***In Montgomery reduction***, the value μ is used to zero w least significant bits of a value a. First, a multiplier m = μa mod 2^w is computed. The value m has at most w bits. ***Adding a + mn will zero w least significant bits of a, and a may be shifted down w bits.*** Since typically L=kw, where k is the number of w-bit words in R; this operation is repeated k times to effect the Montgomery reduction aR^{-1} mod n.

Ex. B at 2:47-53; *see also id.* at Fig. 4; 1:39-46, 5:10-17. Accordingly, MARA's construction should be adopted.

Malikie's construction should be rejected as imprecise and vague. Specifically, it is unclear how adding a multiple of the modulus would "eliminate" the LSW in a manner that results in reduction. To the extent that Malikie intends "eliminate" to cover how Montgomery reduction proceeds as known in the art and described in the '286 Patent, then MARA's construction describes that more precisely.

B. The '062 and '960 Patents

The '960 and '062 Patents[4] are generally directed to methods for performing finite field operations. Ex. D at 4:10-11. The asserted claims are more specifically directed to performing finite field operations (e.g., multiplication, addition, subtraction) on finite field elements represented in a fixed number of machine words. *Id*. at Abstract, 4:11-14. The finite field operation is performed using wordsized operations, where the wordsized operations produce unreduced values. *Id*. at 4:17-22. After completing the wordsized operations, reduction to a specific finite field is then performed. *Id*. at 4:22-23. An example is provided below illustrating

[4] The '960 and '062 Patents share a common specification.

finite field multiplication according to the method disclosed in the ’960 and ’062 Patents.

’062 / ’960 Patents – Finite Field Multiplication
Assumptions: modulus n = 3989; word-size = 2; A = 1289; B = 3456
Goal: calculate A × B mod n
Answer: 1289 × 3456 = 4454784; 4454784 mod 3989 = 3060
Split A and B into word-sized representations:
$A = [12, 89]$
$B = [34, 56]$
Multiply A and B word-by-word to generate intermediate results $P = [P_3, P_2, P_1, P_0]$
$P_3 = [0]$
$P_2 = [12 \times 34] = 408$
$P_1 = [12 \times 56 + 89 \times 34] = 3698$
$P_0 = [89 \times 56] = 4984$
Retain the LSW of each intermediate result (underlined) and add the more significant words (in red) to the more significant intermediate result
$P_3 = [0] \rightarrow [0 + 4] = 4$
$P_2 = [408] = [8 + 36] = [44] \rightarrow [44 + 1] = 45$
$P_1 = [3698] = [98 + 49] = [147] \rightarrow 47$
$P_0 = [4984] = 84$
Full product $P = [P_3, P_2, P_1, P_0]$
$P = [4, 45, 47, 84]$
$P = 4454784$
Reduce P to the finite field (P mod n)
$4454784 \bmod 3989 = 3060$

1. “finite field operation”

MARA’s Construction	Malikie’s Construction
“operation where each operand is a finite field element”	“operation in a finite field”

MARA’s construction is supported by the claim language and the specification, and clarifies the claims for the fact finder. First, MARA’s construction is consistent with the preambles of asserted independent claim 1 of the ’062 Patent and asserted independent claim 3 of the ’960 Patent, which state that the method is “performing a finite field operation on elements of a finite field.” MARA does not contend that the preamble is limiting, however, the term “finite field operation” is used in the body of the claim and should be construed in a manner consistent with how it is used in the preamble.

Second, the specification describes that finite field operations include finite field addition, multiplication, squaring, inversion, and subtraction. Ex. D at Fig. 3; 6:49-60 ("Referring to FIG. 3…[t]he implementation of the protocols 210 requires the use of both elliptic curve operations and finite field operations."), Fig. 5 (listing as finite field operations ("FF Addition, FF Multiplication, FF Squaring, FF Inversion, FF Subtraction"), 7:20-35 ("Referring to FIG. 5, the finite field engine 400 is shown…plurality of finite field operations shown generally at numeral 430 are provided."). Each of these finite field operations takes as input one or more finite field elements. For example, finite field addition adds "two finite field elements"; finite field multiplication "multipl[ies] two elements"; finite field inversion "invert[s] an element." *Id*. at 8:65-67; 9:66-10:1; 12:5-6.

MARA's construction further clarifies that the operands of finite field operations are finite field elements, as opposed to quantities of a different kind such as elliptic curve points. The specification draws a clear distinction between finite field operations, which operate on finite field elements, and elliptic curve operations, which operate on elliptic curve points. For example, the specification describes that each "elliptic curve point consists of two finite field elements. The finite field elements are only operated on directly by the finite field engine." *Id*. at 7:36-41; *see also id.* at 7:20-32 (describing that a finite field engine comprises finite field operations). The specification also confirms that finite field operations and elliptic curve operations are distinct types of operations. *Id*. at 6:52-54 (stating that implementing ECDSA protocol "requires the use of both elliptic curve operations and finite field operations.; *see also id.* at 7:15-16 ("Each elliptic curve operation 320 requires certain finite field operations."), 7:62-67 (states that elliptic curve operations "in turn direct finite field operations."), Figs. 3-4, 7 (listing elliptic curve operations and finite field operations separately); *see also* Ex. A, ¶¶67-69. Thus, the asserted claims are

directed to finite field operations having operands that are finite field elements; while the claims are not directed to elliptic curve operations having operands that are elliptic curve points.

The Court should reject Malikie's construction, which simply rearranges the claim term without further explanation, and therefore fails to "clarify and…to explain what the patentee covered by the claims." *U.S. Surgical Corp.* v. *Ethicon, Inc.* 103 F.3d 1554, 1568 (Fed. Cir. 1997). Furthermore, to the extent Malikie's construction captures elliptic curve operations, whose operands are points on an elliptic curve implemented over an underlying finite field, its construction is contrary to the specification for reasons explained above.

2. "unreduced result"

MARA's Construction	Malikie's Construction
"result without any reduction to a specific finite field or wordsize reduction"	"result of an operation whose bit length has not been lowered"

MARA's construction aligns with the intrinsic evidence, which demonstrates that an unreduced result is a result that is obtained/generated without performing any reduction to a specific finite field or wordsize reduction.

First, asserted independent claim 1 of the '062 Patent recites "obtain a first set of instruction for performing the finite field operation on values representing the elements of the finite field; executing the first set of the instructions to generate an *unreduced result* completing the finite field operation." Asserted independent claim 3 of the '960 Patent recites "completing said non-reducing wordsized operation for each word of said representations to obtain an *unreduced result*." Thus, in the '062 claims the first set of instructions do not perform any type of reduction, and in the '960 claims the non-reducing wordsized operations for each word do not perform any type of reduction.

Second, the specification teaches that there are two types of reduction, reduction to a specific finite field and wordsized reduction. Ex. D at 8:40-49 ("reduction may be specific to a

certain finite field, or a wordsize reduction").[5] In order to generate or obtain an "unreduced result," neither of those types of reduction can be performed as part of the "first set of instructions" or the "non-reducing wordsized operation for each word." Thus, the plain meaning of "unreduced result" is a result obtained/generated *without reduction to a specific finite field or wordsize reduction*. If either of those types of reductions is performed, then the result that is obtained would *not* be an unreduced result.

Third, during the prosecution of both patents, the applicant added the term "unreduced result" to distinguish over prior art that teaches reducing intermediate results before completing an operation, whereas the claims cover completing the operation without reducing intermediate results and performing reduction only after obtaining the full result.

During prosecution of the '960 Patent, the applicant distinguished cited prior art Vanstone on the basis that Vanstone "taught the generation of partial products during multiplication and the reduction prior to accumulation." Ex. I at 844; *see also* Section I.A.2, *supra* (explaining that multiplication may be performed by first multiplying portions of the multiplicands to generate partial products, followed by accumulating the partial products to obtain the full product); Ex. A, ¶32. The applicant argued that the alleged invention "distinguished over this by accumulating the partial products…and subsequently, upon completion of the accumulation performing a modular reduction." Ex. I at 844. Similarly, the applicant distinguished another cited reference Dworkin, arguing that Dworkin teaches "intermediate reduction of the partial products," whereas the claims "requir[e] completion of those operations prior to reduction." *Id.* Moreover, during the examiner

[5] A POSA would understand that reduction "specific to a certain finite field" refers to reducing an input *a* to the modulus of the finite field *n* (i.e., calculating a mod n). Ex. A, ¶73. Wordsize reduction refers to "lower[ing] the length of the result to the appropriate word length of the underlying field." Ex. D at 8:44-47. Dr. Koc's declaration provides an example to illustrate the difference between these two types of reduction. Ex. A, ¶74.

interview prior to issuance, the "applicant proposed language to distinguish from the prior art of record by including the limitations of performing computations without reduction and the reductions occur after the results are obtained." *Id.* at 724. The applicant then amended the claims to add the limitation "unreduced result." *Id.* at 715. The claims subsequently issued. In the Notice of Allowance, the examiner noted that the cited prior art "fails to disclose performing finite field operations by performing non-reducing computations and then a reduction at the end. [Instead,] Dworkin performs reduction at each step, and does not just perform on [*sic*] reduction at the end." *Id.* at 680.

During the prosecution of the '062 Patent, the applicant amended the pending claims to add "unreduced result" in response to the examiner's rejection over Dworkin. Ex. J at 1608. In its accompanying remarks, the applicant argued that the claims do not "reduc[e] fully (i.e., at each step), as was done prior." *Id.* at 1613. The examiner maintained the rejection over Dworkin. In response, the applicant amended the claims further to add the limitation "generate an unreduced result completing the finite field operation." *Id.* at 1581. The applicant stressed again that the claims do not cover operations that perform reduction on intermediate results. *Id.* ("Applicant has therefore recognized that by reducing the result of the routine when performing a finite field operation, rather than reducing fully (i.e. at each step), as was done prior…"). Thus, the prosecution history demonstrates that the applicant intended "unreduced result" to mean that the result is obtained/generated without intermediate reduction, which the specification explains to include reduction to a specific finite field or wordsize reduction.

Malikie's construction should be rejected at least because it is contrary to the file history. As discussed above, the applicant added the term "unreduced result" to distinguish over prior art on the basis that the prior art teaches performing a finite field operation (multiplication) with

intermediate reduction of partial products, whereas the claimed method does not perform any intermediate reduction of partial products. Therefore, the term "unreduced result" is used to define *how* the result is generated (i.e., the result is generated by executing instructions or wordsize operations that do not include any reduction step). Malikie's construction does not address *how* the result is generated, and therefore, does not prohibit the instructions from including a reduction step. This is contrary to the applicant's statements in the file history.

Malikie's construction is also vague in that a POSA would not be able to determine with reasonable certainty what the bit length of the result should be compared with to determine that its bit length has not been lowered. Ex. A, ¶76. For example, the specification describes a method for finite field multiplication generating an "unreduced product." Ex. D at 11:44-65. The method involves multiplying two finite field elements, generating 32 partial products, and then accumulating the partial products to generate the unreduced product. *Id.* It is unclear under Malikie's construction whether the bit-length of the product should be compared to either one or both of the inputs, any or all of the partial products, or some other unspecified quantity.

3. "reduced result"

MARA's Construction	Malikie's Construction
No construction needed. Plain and ordinary meaning.	"result of an operation whose bit length has been lowered"

The claims explicitly define what a "reduced result" is, and therefore, no further construction is needed. *Biosonix, LLC* v. *Hydrowave, LLC*, 230 F.Supp.3d 598, 605 (Fed. Cir. 2017) ("This court is simply not required to rewrite the limitations…that is already defined in substantial detail within a claim."). Specifically, in the claims of the '062 Patent, the reduced result is a result generated by "performing a ***modular reduction for a specific finite field***…on the unreduced result." In the claims of the '960 Patent, the reduced result is a result obtained by

"performing a ***specific modular reduction*** of said unreduced result ***to reduce*** said unreduced result ***to that of a field element of said finite field***." Thus, in both patents, the claims specify that modular reduction is performed on the unreduced result to generate the reduced result (i.e., a mod $n = b$, wherein a is the unreduced result, n is the modulus, b is the reduced result).

The Court should reject Malikie's proposal because it is contrary to the claim language. The claims specify that the reduced result is obtained by performing modular reduction. However, modular reduction does not always lower the bit-length of the unreduced result. Ex. A, ¶76. Consider a decimal example in a finite field consisting integers 0-18 (*i.e.*, modulus n = 19). Subtracting 6 (finite field element) from 5 (finite field element) generates the unreduced result -1. Performing modular reduction on the unreduced result -1 yields a result of 18 (i.e., -1 mod 19 = 18). The result of 18 is generated by performing modular reduction, however, its bit-length is *more* than that of the unreduced result -1.[6] Ex. A, ¶76.

C. The '827 and '370 Patents

The '827 and '370 Patents[7] relate to the prior art Elliptic Curve Digital Signature Algorithm ("ECDSA"), and more specifically, to public key recovery. Ex. E at 1:24-34; 2:28-41. ECDSA dictates that the signer selects a long term private key d and computes a long term public key Q using $Q = dG$, where G is a generator. *Id.* at 2:10-16, 2:28-41. The signer sends a message M, and creates a signature, which is comprised of a pair of integers (r, s). *Id.* at 2:42-49. The signing process includes "comput[ing] a point R=kG that has coordinates (x, y)." *Id.* at 2:49-63. The verifier can take the message M, the public key Q, and the signature comprising (r, s) and verify if

[6] Under a typical method to store negative integers in binary, –1 is stored as 11 and 18 is stored as 010010. Ex. A, ¶76.

[7] The '827 and '370 Patents share a common specification.

the message was created by the signer. *Id*. at 2:64-3:4.

The Background of the Invention further describes that ECDSA signatures can be generated very quickly, while ECDSA signature verification is relatively slower. *Id*. at 4:16-25. Accordingly, the specification contends there is a need to increase the efficiency of the signature verification computation. *Id*. at 4:25-33.

The last part of the specification is directed to the verifier recovering the signer's public key *Q* from the ECDSA signature to avoid looking up the public key in a database. *Id*. at 15:15-40. This public key recovery process, including a particular equation used to recover the public key *Q*, is shown in Figure 14. All asserted claims in the '827 and '370 Patents are directed to this specific public key recovery process.

1. "which provides for an accelerated verification of the received signature" ['370]

MARA's Construction	Malikie's Construction
Not limiting. In the alternative, indefinite.	Not indefinite – plain and ordinary meaning

The term "which provides for an accelerated verification of the received signature" is not limiting under Federal Circuit precedent. To the extent it is limiting, the term is indefinite.

The Federal Circuit has held certain claim terms to be non-limiting because they "merely express[ed] a purpose" and "only state[d] an intended result of the claimed method." *Bristol-Myers Squibb Co.* v. *Ben Venue Labs., Inc.*, 246 F.3d 1368, 1375 (Fed. Cir. 2001). "Claim language without any bearing on the claimed methods should be deemed non-limiting when it does not result in 'a manipulative difference in the steps of the claim.'" *Teva Pharms. USA, Inc.* v. *Sandoz Inc. (In re Copaxone Consol. Cases)*, 906 F.3d 1013, 1023 (Fed. Cir. 2018) (quoting *Bristol-Myers Squibb Co.*, 246 F.3d at 1376). The Court in *Teva Pharms.* held that the phrase "the regimen being sufficient to reduce the frequency of relapses in the patient" was non-limiting

because it "does not change the claimed method or require any additional required structure or condition for the claims." *Id*. at 1023.

Here, the language "which provides an accelerated verification of the received signature" is stating an intended result that flows from performing the recited method steps of "*receiving*…an electronic message including a signature…; *receiving*…a first elliptic curve point associated with a signature component…; "*recovering*…the omitted public key of the signer…wherein the recovering comprises computing $Q = r^{-1}(sR – eG)$…; and *verifying*…the received signature using the recovered public key…" The limitation does not change the claimed method or require any additional required structure or condition for the claim. Accordingly, the phrase "which provides an accelerated verification of the received signature" is non-limiting.

To the extent the phrase is deemed to place any additional limit on the claim beyond performing the recited method steps, a POSA could not determine with reasonable certainty the scope of the invention. Ex. A, ¶79. First, if performing the recited method steps is not sufficient to "provide an accelerated verification of the received signature," it is unclear what more is required to yield the result of "accelerated verification." *Id*. Second, if "accelerated verification" is deemed to provide a limitation beyond the recited method steps, then the term, "when read in light of the specification and prosecution history, must provide objective boundaries for those of skill in the art." *HZNP Meds. LLC* v. *Actavis Labs. UT, Inc.*, 940 F.3d 680, 695 (Fed. Cir. 2019). Here, no such objective boundary exists to determine whether a verification performed according to the recited method steps is accelerated as opposed to not accelerated. *Id.*, ¶¶79-81. Accordingly, if found to be limiting, the phrase "which provides an accelerated verification of the received signature" is indefinite.

2. "the electronic message omits a public key of a signer" ['370]

MARA's Construction	Malikie's Construction
"the electronic message does not include any representation of the public key of the signer"	Plain and ordinary meaning

MARA's construction is supported by the claim language and the prosecution history.

First, the surrounding claim language confirms that omitting the public key not only refers to omitting the exact public key (Q) but also refers to omitting a representation of the public key Q (e.g., a compressed version of the public key Q). Specifically, the claims recite recovering the public key Q using a specific equation: $Q = r^{-1}(sR - eG)$. If the message were to include a representation of the public key Q such that the public key Q could be obtained from the representation, then there would be no need to recover the public key Q using the specific equation recited in the claim. Moreover, the specification confirms that the purpose of the specific equation is to be able to recover the public key Q from only the signature. Ex. F at 15:46-47 ("Yet another application of this technique is to provide efficient recovery of the public key Q from only the ECDSA digital signature as shown in FIG. 14."), 15:56-57 ("To avoid [looking up the public key in a database], it would beneficial to be able to recover the public key Q from the signature.").

Second, the applicant's arguments during the prosecution history confirm that omission of the public key Q means omitting any representation of the public key Q. In a July 20, 2016 Non-Final Rejection, the examiner stated that "[c]laims 1, 46 & 53 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ono (US 20020172356) in view of Gallant (US2002/0044649). Ex. K at 333. Even though Ono did "not teach explicitly *generating, using data processing apparatus, a public key of the signer based on the elliptic curve point and the signature*," the examiner held that "Gallant teaches *generating, using data processing apparatus, a public key of the signer based on the elliptic curve point and the signature.* [0019] The sender sends to the recipient a message

including m, s, and R and the signature is verified by computing the value R.sup.1=(sP-eQ) which should correspond to R. If the computed values correspond then the signature is verified. From the cited equation it is obvious to a skilled person in the art that that Public key Q can be derived (generated) on the basis of signature (signature component (s)) and the elliptic curve point (P).” *Id*. at 334.

In response, the applicant submitted an October 20, 2016 Amendment adding the limitation “wherein the electronic message omits any public key of a signer.” *Id*. at 312. In order to overcome the rejection, the applicant argued that Gallant does not teach omitting a public key of a signer because it discloses sending a short term public key R that is used to compute the signer’s long term public key Q:

> “[f]or the teaching of the generation, the Office Action offers the verification equation disclosed in *Gallant* and states that the senders corresponding public key can be solved for using the equation. However, Applicant respectfully submits that the equation disclosed in *Gallant* uses the short-term public key included in the signed message. *Gallant* states, ‘The sender sends to the recipient a message including m, s, and R and the signature is verified by computing the value R.sup.1=(sP-eQ) which should correspond to R.’ ¶ 0019. ***In order to solve for the public key Q, the short term public key R must be included in the message. Thus, Applicant respectfully submist [sic] that the cited combination fails to teach or suggest the received messaging omitting any public keys of the signer***”

Id. at 320. Thus, in the above argument, the applicant made clear that omitting a public key of the signer is not limited to omitting the exact public key Q but also requires omitting a representation of the public key Q (which can be used to compute the public key Q).

3. “verifying that the second elliptic curve point Q represents the public key of the signer” [’827]

MARA’s Construction	Malikie’s Construction
“verifying that the second elliptic curve point Q represents the second elliptic curve point Q”	Plain and ordinary meaning

Claim 2 requires "verifying that the second elliptic curve point *Q* represents the public key of the signer." The antecedent basis for "the second elliptic curve point *Q*" and "the public key of the signer" are in claim 1, which states "generating … a public key of the signer" and that "the public key of the signer comprises a second elliptic curve point *Q*, generating the public key of the signer comprises computing *Q* = …" Thus, in claim 1 and claim 2, both "the second elliptic curve point *Q*" and "the public key of the signer" are the value computed using the equation $Q = r^{-1}(sR - eG)$. Neither claim 1 nor claim 2 identifies a public key of the signer that is received (as opposed to computed). Thus, while perhaps nonsensical, claim 2 literally requires verifying that second elliptic curve point *Q* represents the second elliptic curve point *Q*. *Chef Am., Inc.* v. *Lamb-Weston, Inc.*, 358 F.3d 1371, 1374 (Fed. Cir. 2004) ("Even a nonsensical result does not require the court to redraft the claims of the [patent].").

D. The '961 Patent

The '961 Patent relates to the generation of a cryptographic key for use in a cryptographic function. Specifically, the '961 Patent purports to solve the issue of modulo bias in the generation of a key k for use in a cryptographic protocol such as DSA. Ex. G at 3:1-3. Thus, the claim terms should be construed to have their plain meaning in the context of cryptography and cryptographic signature protocols such as DSA. DSA was implemented in a prior art standard known as the Digital Signature Standard (DSS) published by the National Institute of Standards and Technology ("NIST") in the 1990s. *Id*. at Abstract, 2:36-38.

The '961 Patent describes how a version of DSS known as FIPS 186-2, published in 2000, generates an ephemeral cryptographic key, k. *Id*. at 2:36-46. Specifically, FIPS 186-2 generates a seed value (SV) from a random number generator, which is then hashed by a SHA-1 hash function to yield a bit string of predetermined length, typically 160 bits. *Id*. at 2:40-43. The bit string represents an integer between 0 and 2^{160}-1. *Id*. at 2:43-44. Because this integer can be

greater than the prime q, however, DSS reduces the integer by computing "SHA-1(seed) mod q," ensuring that the value of the key k will fall within the desired range of 0 to q-1. *Id*. at 2:44-46.

The '961 Patent admits that it was known that using a "mod q" operation could introduce modulo bias in favor of smaller values. *Id*. at 2:53-55. The '961 Patent purports to eliminate this bias by using a well-known technique, whereby hashed seed values that are greater than q are rejected, and new hashed seed values are produced until one less than q is obtained. *Id*. at 4:11-17.

Figure 2 illustrates the rejection sampling technique described in the '961 patent. The method "originates by obtaining a seed value (SV) from the random number generator 26." *Id*. at 3:64-66. The seed value is then provided to hash function 28 to produce an output—H(seed)—of the correct length. *Id*. at 4:6-10. The hashing function is used "to yield a bit string of predetermined length, typically 160 bits." *Id*. at 2:40-43, 3:38-41. This is the same value represented as "SHA-1(seed)" in the DSA. *Id*., 2:40-46. The remaining steps in Figure 2 perform the well-known rejection sampling technique, which checks if H(seed) is less than q, where the maximum desired value is q – 1. If the answer is yes, H(seed) is accepted and used as the cryptographic key k. *Id*. at 4:11-13. If no, the value H(seed) "is rejected" and the process starts over with generation of a new seed value from the RNG. *Id*., 4:13-17. This loop will continue indefinitely "until a satisfactory value is obtained." *Id*.

1. "random number generator"

MARA's Construction	Malikie's Construction
"a system or algorithm that generates a random value"	"computer instructions capable of generating values according to a uniform random probability distribution"

The dispute concerning "random number generator" is whether it generates random values. The claim language, specification, and prosecution history all confirm that in the context

of the '961 Patent, the plain meaning of "random number generator" is "a system or algorithm that generates a random value." Malikie's construction, which does not require the "random number generator" to generate random values and includes both random and pseudorandom number generators, is overly broad and should be rejected.

First, the claim language, requires, *inter alia*, "generating a seed value SV from ***a random number generator***," where the seed value is used in subsequent calculations to create a "key k for use in performing said ***cryptographic function***." Ex. G at Claim 1. In the context of cryptography, a POSA would understand the distinction between random number generators ("RNGs") and pseudorandom number generators ("PRNGs"). Ex. A, ¶¶82-87. Specifically, a POSA would understand that RNGs generate random values, while PRNGs generate deterministic values that can appear to be random but are ***not actually random***. *Id.*

This distinction (*i.e.*, random values v. deterministic values that can appear random) is explicitly reflected in the claims of the '961 Patent. While claim 1 requires generating an input to a hash function (*i.e.*, the seed value) using a ***random*** number generator, claim 7 requires calculating an input to a hash function by incrementing a rejected output using a ***deterministic*** function. Ex. G at Claim 1 ("generating a seed value…from an [RNG and] "performing a hash function H() on said seed value SV…."), Claim 7 ("if said output is rejected, said output is incremented by a ***deterministic function*** and a hash function is performed on said incremented output."). Claims 1 and 7 demonstrate that the inventors of the '961 Patent were aware of the distinction between random and deterministic, and drafted their claims accordingly. The claim language thus supports MARA's construction, while confirming that Plaintiffs' construction should be rejected because it eliminates the distinction that the inventors drew between random and deterministic values. *CAE Screenplates Inc.* v. *Heinrich Fiedler GmbH & Co. KG*, 224 F.3d 1308, 1317 (Fed. Cir. 2000)

("We must presume that the use of these different terms in the claims connotes different meanings.") (internal citations omitted).

Second, the specification supports MARA's construction. The specification describes that the '961 Patent purports to provide an improved key generation algorithm for use with protocols such as DSA, which was implemented in a prior art standard known as the Digital Signature Standard ("DSS"). Ex. G at Abstract, 2:32-61. The earliest version of DSS, which published in 1994, distinguishes between randomly generated and pseudorandomly generated integers. Ex. P at 2068 ("x = a randomly ***or*** pseudorandomly generated integer… k = a randomly ***or*** pseudorandomly generated integer,"), 2075 ("Any implementation of the DSA requires the ability to generate random ***or*** pseudorandom integers…."); Ex. A, ¶86. In view of the distinction between random and pseudorandom described in DSS, "random number generator" in claim 1 should not be construed to also encompass pseudorandom number generators, as Malikie's construction does. Ex. A, ¶¶82-87.

Third, the prosecution history supports MARA's construction because the applicant repeatedly asserted that the seed value generated by the random number generator is a random value. *Eye Therapies, LLC* v. *Slayback Pharma, LLC*, 141 F.4th 1264, 1269 (Fed. Cir. 2025) (explaining that statements made to an examiner "do inform the claim construction") (internal citations omitted); *Phillips* v. *AWH Corp.*, 415 F.3d 1303, 1317 (Fed. Cir. 2005) ("[T]he prosecution history can often inform the meaning of the claim language by demonstrating how the inventor understood the invention."). Indeed, the applicant consistently distinguished prior art on the basis that the random number generator of claim 1 generates a random seed value. Ex. L at 1227 ("Step (c) of claim 1 involves determining whether the output is less than q, where the output is a result of a hash on ***a seed number that is chosen at random***…. In Schneier…*k* is not an output

that is a result of a hash on ***a seed number chosen at random***.") (bolded emphasis added); *see also id*. at 1228 ("Backal does not teach…where the output is provided by performing a hash function on a ***seed value generated at random***.").

In IPR2019-00923, the prior owner of the 961 Patent likewise distinguished claim 1 from the prior art on the basis that the claimed random number generator generates random seed values:

> The petition fails to explain why Menezes' technique for accepting or rejecting a "random integer" based on whether it lies within a specific range of randomly generated seed value… ***Claim 1 specifically provides distinct steps for generating a seed value from a random number generator***… Yet, the petition does not apply Menezes' teaching against ***the step in claim 1 where a random number is actually generated (i.e., "generating a seed value from a random number generator")***; instead, Petitioners map Menezes to the subsequent steps in claim 1 for accepting or rejecting the hash of the randomly generated seed value.

Ex. Q at 41-42 (Aug. 09, 2019); *see also id*. at 46 ("Despite the failure of any reference cited in Ground 2 to teach repetition of a method that includes ***randomly generating a seed value*** and hashing that value…."), 47 ("[A]s noted above, neither Menezes nor DSS nor Schneier teach that the ***random generation of a seed value*** and hashing of that seed value"), 48 ("[T]he petition's proposal to repeat both ***random seed value generation*** and hashing…."); *see also Aylus Networks, Inc.* v. *Apple Inc.*, 856 F.3d 1353, 1361 (Fed. Cir. 2017) ("[S]tatements made by a patent owner during an IPR proceeding can be considered during claim construction…."). The prosecution and IPR statements confirm what is clear from the claim language and specification—the claimed "random number generator" generates random values, including a random seed value.

Finally, the extrinsic evidence confirms that MARA's construction is correct. Consistent with the fact that DSS distinguishes between RNGs that produce random values and PRNGs that can produce deterministic values, the prior art Handbook for Applied Cryptography contains separate definitions for "random bit generator" and "pseudorandom bit generator," and clearly states that "the output" of a pseudorandom bit generator is "*not* random." Ex. R at 628 (emphasis

in original).[8] Thus, extrinsic evidence in the field of cryptography supports MARA's construction and demonstrates that Plaintiffs' construction is overly broad.

2. "seed"

MARA's Construction	Malikie's Construction
"a random value that is used as the starting value for a cryptographic key generation function"	"a value obtained from a random number generator that is used to as the starting value for a cryptographic key generation function"

The parties dispute whether the "seed" value generated from the "random number generator" is a random value.[9] Even if the Court accepts Plaintiffs' position that a "random number generator" can generate values that are not random, the intrinsic evidence demonstrates that the "seed" generated by the RNG in claim 1 is a random value.

First, the plain language of the claim compels adoption of MARA's construction. Claim 1 requires "generating a seed value SV from a ***random*** number generator." Ex. G, claim 1. Under Malikie's constructions, the "random number generator" does not need to generate a random value, and the "seed value" likewise need not be a random value. Thus, Malikie's constructions of those terms render the term "random" superfluous. *Intel Corp.* v. *Qualcomm Inc.*, 21 F.4th 801, 810 (Fed. Cir. 2021) ("It is highly disfavored to construe terms in a way that renders them void, meaningless, or superfluous.") (quoting *Wasica Finance GmbH* v. *Continental Automotive Systems, Inc.*, 853 F.3d 1272, 1288 n.10 (Fed. Cir. 2017)).

[8] The terms "random bit generator," and "random number generator" are functionally synonymous for purposes of the '961 patent, as both produce random numbers. Ex. A, ¶86; Ex. R at MARA_0000628.

[9] The parties agree that a "seed" is a "starting value for a cryptographic key generation function." Plaintiffs' further inclusion of "from a random number generator" is unnecessary given that claim 1 explicitly states that the seed value is generated "from a random number generator."

Second, construing "seed" to include deterministic values has no support in the specification. The '961 Patent's specification parrots the plain language of the claims, which, as noted above, clearly indicates that the seed value is random because it is generated from a random number generator. Ex. G at 3:64-66 ("[A] first method of generating a key, k, originates by obtaining a seed value (SV) from the random number generator 26."), 4:37-39 ("The random number generator 26 generates a seed value SV").

Third, and as explained above, the applicant repeatedly asserted during prosecution and a subsequent IPR that the "seed value" generated from a "random number generator" is a random value. Ex. L at 1227-1228; Ex. Q at 41-42, 46, 47, 48. These statements confirm what is clear from the plain language of claim 1—the seed value is a random value. Accordingly, MARA's construction should be adopted.

3. "The method of claim 1 wherein if said output is rejected, said output is incremented by a deterministic function and a hash function is performed on said incremented output to produce a new output; a determination being made as to whether said new output is acceptable as a key."

MARA's Construction	Malikie's Construction
Indefinite	Not indefinite – plain and ordinary meaning

Claim 1 requires, *inter alia*, (1) generating a seed value SV from a random number generator (RNG) and (2) performing a hash function H( ) on SV to provide an output H(SV). Claim 1 further requires "rejecting said output H(SV) as said key if said value is not less than said order q" and repeating said method "if said output H(SV) is rejected." Thus, claim 1 requires repeating the steps of generating a seed value SV from an RNG and performing a hash on SV to provide a new output H(SV) if a first output H(SV) is rejected, as illustrated in Figure 2 and described in the corresponding specification text. Ex. G at 4:11-17, Fig. 2.

Claim 7 sets forth a different set of steps to perform if output H(SV) is rejected. Specifically, claim 7 requires that "if said output is rejected," which refers to output H(SV), "said output is incremented by a deterministic function and a hash function is performed on said incremented output to produce a new output; a determination being made as to whether said new output is acceptable as a key." The different steps that claims 1 and 7 require if output H(SV) is rejected are summarized below:

Steps after Rejection	**Claim 1**	**Claim 7**
Step 1	Generate new seed value (SV) from an RNG	Increment output H(SV) by a deterministic function
Step 2	Hash function H() is performed on new seed value (SV) to generate a new output H(SV)	Hash function H() is performed on incremented output to produce a new output
Step 3	New output H(SV) is accepted as a key if less than order q and rejected if not less than order q	A determination is made whether the new output is acceptable as a key

Claim 7 depends from claim 1, and therefore requires performing all the steps of claim 1 in addition to the steps in claim 7. *Monsanto Co.* v. *Syngenta Seeds, Inc.*, 503 F.3d 1352, 1359 (Fed. Cir. 2007) ("[C]laims in dependent form include all the limitations of the claim incorporated by reference into the dependent claim"); *see also* 35 U.S.C. § 112(d). In other words, claim 7 requires both repeating the steps of claim 1 to produce a new output H(SV) if the first output H(SV) is rejected, ***and*** performing the steps in claim 7 to produce a new output if the first output H(SV) is rejected. Claim 7, however, fails to provide a POSA with reasonable certainty regarding how to perform the two different sets of steps together to produce a new output H(SV) (claim 1) and/or a new output (claim 7) and assess whether they are acceptable as keys. Ex. A, ¶¶ 88-93. For example, it is not reasonably certain whether, after rejecting output H(SV), claim 7 requires (1) producing only one new output H(SV) by combining both sets of steps in some undisclosed way; (2) producing only one new output H(SV) by hashing a new seed value SV generated from

an RNG; (3) producing only one new output by hashing an incremented rejected output; or (4) producing two different outputs to generate two different candidate keys. *Id*.

Nor does the specification provide guidance to resolve the ambiguity in claim 7. Indeed, the specification only describes the different set of steps in claim 7 as an alternative—it does not describe how the steps after rejection in claim 1 and claim 7 could be performed together. *Compare* Ex. G at Fig. 2 (hashing a second SV generated by an RNG after output H(SV) is rejected), *with* Fig. 3 (hashing an incremented output after output H(SV) is rejected); *see also id*. at 4:50-52 ("Upon rejection, the random number generator may generate a new value as disclosed in FIG. 2 ***or*** may increment the seed value as disclosed in FIG. 3."). Thus, the specification's description of them as alternative steps is contrary to the plain language of claim 7, which requires that both the steps in claim 1 and claim 7 be performed given claim 7's dependency from claim 1. Thus, the specification fails to provide reasonable certainty concerning how to perform the method of claim 7 if output H(SV) is rejected. Ex. A, ¶¶ 92-93.

Malikie's construction—plain meaning—does nothing to resolve the ambiguity in the claim language or otherwise provide reasonable certainty regarding how to perform the method of claim 7 if output H(SV) is rejected. Thus, claim 7 is indefinite.

III. CONCLUSION

MARA respectfully requests that the Court adopt its proposed constructions.

Dated: December 17, 2025

Respectfully Submitted,

PAUL, WEISS, RIFKIND, WHARTON & GARRISON LLP

/s/ Anish Desai
Anish Desai
Elizabeth S. Weiswasser
Ian A. Moore
Tom Yu
Thomas Macchio
Paul, Weiss, Rifkind, Wharton & Garrison LLP
1285 Sixth Avenue
New York, NY 10019
Telephone: (212) 373-3000

Christopher M. Pepe
Priyata Patel
W. Sutton Ansley
Eric C. Westerhold
Paul, Weiss, Rifkind, Wharton & Garrison LLP
2001 K Street NW
Washington, DC 20006
Telephone: (202) 223-7300

Steve Wingard
State Bar No. 00788694
swingard@scottdoug.com
Robert P. Earle
State Bar No. 241245566
rearle@scottdoug.com
Stephen L. Burbank
State Bar No. 24109672
sburbank@scottdoug.com
Scott Douglass & McConnico LLP
303 Colorado Street, Suite 2400
Austin, TX 78701-3234
Telephone: (512) 495-6300
Facsimile: (512) 495-6399

Counsel for Defendant MARA Holdings, Inc.

CERTIFICATE OF SERVICE

The undersigned attorney hereby certifies that on December 17, 2025, I caused correct copies of the foregoing document to be served via email on all counsel of record.

/s/ Anish Desai
Anish Desai